$\mathcal{N}$ATTACK: Learning the Distributions of Adversarial Examples for an Improved Black-Box Attack on Deep Neural Networks

Yandong Li [*1] **Lijun Li** [*1] **Liqiang Wang** [1] **Tong Zhang** [2] **Boqing Gong** [3]

Abstract

Powerful adversarial attack methods are vital for understanding how to construct robust deep neural networks (DNNs) and thoroughly testing defense techniques. In this paper, we propose a black-box adversarial attack algorithm that can defeat both vanilla DNNs and those generated by various defense techniques developed recently. Instead of searching for an "optimal" adversarial example for a benign input to a targeted DNN, our algorithm finds a probability density distribution over a small region centered around the input, such that a sample drawn from this distribution is likely an adversarial example, without the need of accessing the DNN's internal layers or weights. Our approach is *universal* as it can successfully attack different neural networks by a single algorithm. It is also *strong*; according to the testing against 2 vanilla DNNs and 13 defended ones, it outperforms state-of-the-art black-box or white-box attack methods for most test cases. Additionally, our results reveal that adversarial training remains one of the best defense techniques, and the adversarial examples are not as transferable across defended DNNs as them across vanilla DNNs.

1. Introduction

This paper is concerned with the robustness of deep neural networks (DNNs). We aim at providing a strong adversarial attack method that can universally defeat a variety of DNNs and associated defense techniques. Our experiments mainly focus on attacking the recently developed defense methods, following (Athalye et al., 2018). Unlike their work, however, we do not need to tailor our algorithm to various forms for tackling different defenses. Hence, it may generalize better to new defense methods in the future. Progress on powerful adversarial attack algorithms will significantly facilitate the research toward more robust DNNs that are deployed in uncertain or even adversarial environments.

Szegedy et al. (2013) found that DNNs are vulnerable to adversarial examples whose changes from the benign ones are imperceptible and yet can mislead DNNs to make wrong predictions. A rich line of work furthering their finding reveals more worrisome results. Notably, adversarial examples are *transferable*, meaning that one can design adversarial examples for one DNN and then use them to fail others (Papernot et al., 2016a; Szegedy et al., 2013; Tramèr et al., 2017b). Moreover, adversarial perturbation could be *universal* in the sense that a single perturbation pattern may convert many images to adversarial ones (Moosavi-Dezfooli et al., 2017).

The adversarial examples raise a serious security issue as DNNs become increasingly popular (Silver et al., 2016; Krizhevsky et al., 2012; Hinton et al., 2012; Li et al., 2018; Gan et al., 2017). Unfortunately, the cause of the adversarial examples remains unclear. Goodfellow et al. (2014b) conjectured that DNNs behave linearly in the high dimensional input space, amplifying small perturbations when their signs follow the DNNs' intrinsic linear weights. Fawzi et al. (2018) experimentally studied the topology and geometry of adversarial examples and Xu et al. (2019) provide the image-level interpretability of adversarial examples. Ma et al. (2018) characterized the subspace of adversarial examples. Nonetheless, defense methods (Papernot et al., 2015; Tramèr et al., 2017a; Rozsa et al., 2016; Madry et al., 2018) motivated by them were broken in a short amount of time (He et al., 2017; Athalye et al., 2018; Xu et al., 2017; Sharma & Chen, 2017), indicating that better defense techniques are yet to be developed, and there may be unknown alternative factors that play a role in the DNNs' sensitivity.

Powerful adversarial attack methods are key to better understanding of the adversarial examples and for thorough testing of defense techniques.

In this paper, we propose a black-box adversarial attack algorithm that can generate adversarial examples to defeat both vanilla DNNs and those recently defended by various

[*]Equal contribution [1]University of Central Florida [2]Hong Kong University of Science and Technology [3]Google. Correspondence to: Yandong Li <lyndon.leeseu@outlook.com>, Boqing Gong <BoqingGo@outlook.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

techniques. Given an arbitrary input to a DNN, our algorithm finds a probability density over a small region centered around the input such that a sample drawn from this density distribution is likely an adversarial example, without the need of accessing the DNN's internal layers or weights — thus, our method falls into the realm of black-box adversarial attack (Papernot et al., 2017; Brendel et al., 2017; Chen et al., 2017; Ilyas et al., 2018).

Our approach is *strong*; tested against two vanilla DNNs and 13 defended ones, it outperforms state-of-the-art black-box or white-box attack methods for most cases, and it is on par with them for the remaining cases. It is also *universal* as it attacks various DNNs by a single algorithm. We hope it can effectively benchmark new defense methods in the future — code is available at `https://github.com/Cold-Winter/Nattack`. Additionally, our study reveals that adversarial training remains one of the best defenses (Madry et al., 2018), and the adversarial examples are not as transferable across defended DNNs as them across vanilla ones. The latter somehow weakens the practical significance of white-box methods which otherwise could fail a black-box DNN by attacking a substitute.

Our optimization criterion is motivated by the natural evolution strategy (NES) (Wierstra et al., 2008). NES has been previously employed by Ilyas et al. (2018) to estimate the gradients in the projected gradient search for adversarial examples. However, their algorithm leads to inferior performance to what we proposed (cf. Table 1). This is probably because, in their approach, the gradients have to be estimated relatively accurately for the projected gradient method to be effective. However, some of the neural networks $F(x)$ are not smooth, so that the NES estimation of the gradient $\nabla F(x)$ is not reliable enough.

In this paper, we opt for a different methodology using a constrained NES formulation as the objective function instead of using NES to estimate gradients as in Ilyas et al. (2018). The main idea is to smooth the loss function by a probability density distribution defined over the ℓ_p-ball centered around a benign input to the neural network. All adversarial examples of this input belong to this ball[1]. In this frame, assuming that we can find a distribution such that the loss is small, then a sample drawn from the distribution is likely adversarial. Notably, this formulation does not depend on estimating the gradient $\nabla F(x)$ any more, so it is not impeded by the non-smoothness of DNNs.

We adopt parametric distributions in this work. The initialization to the distribution parameters plays a key role in the run time of our algorithm. In order to swiftly find a good initial distribution to start from, we train a regression neural network such that it takes as input the input to the target DNN to be attacked and its output parameterizes a probability density as the initialization to our main algorithm.

Our approach is advantageous over existing ones in multiple folds. First, we can designate the distribution in a *low-dimensional* parameter space while the adversarial examples are often high-dimensional. Second, instead of questing an "optimal" adversarial example, we can virtually draw an *infinite* number of adversarial examples from the distribution. Finally, the distribution may speed up the adversarial training for improving DNNs' robustness because it is more *efficient* to sample many adversarial examples from a distribution than to find them using gradient based optimization.

2. Approach

Consider a DNN classifier $C(x) = \arg\max_i F(x)_i$, where $x \in [0,1]^{\dim(x)}$ is an input to the neural network $F(\cdot)$. We assume softmax is employed for the output layer of the network and let $F(\cdot)_i$ denote the i-th dimension of the softmax output. When this DNN correctly classifies the input, *i.e.*, $C(x) = y$, where y is the groundtruth label of the input x, our objective is to find an adversarial example x_{adv} for x such that they are imperceptibly close and yet the DNN classifier labels them distinctly; in other words, $C(x_{adv}) \neq y$. We exclude the inputs for which the DNN classifier predicts wrong labels in this work, following the convention of previous work (Carlini & Wagner, 2017).

We bound the ℓ_p distance between an input x and its adversarial counterparts: $x_{adv} \in S_p(x) := \{x' : \|x - x'\|_p \leq \tau_p\}, p = 2 \text{ or } \infty$. We omit from $S_p(x)$ the argument (x) and the subscript p when it does not cause ambiguity. Let $\text{proj}_S(x')$ denote the projection of $x' \in \mathbb{R}^{\dim(x)}$ onto S.

We first review the NES based black-box adversarial attack method (Ilyas et al., 2018). We show that its performance is impeded by unstable estimation of the gradients of certain DNNs, followed by our approach which does not depend at all on the gradients of the DNNs.

2.1. A Black-box Adversarial Attack by NES

Ilyas et al. (2018) proposed to search for an optimal adversarial example in the following sense,

$$x_{adv} \leftarrow \arg\min_{x' \in S} f(x'), \tag{1}$$

given a benign input x and its label y correctly predicted by the neural network $F(\cdot)$, where S is a small region containing x defined above, and $f(x')$ is a loss function defined as $f(x') := F(x')_y$. In (Ilyas et al., 2018), this loss is minimized by the projected gradient method,

$$x_{t+1} \leftarrow \text{proj}_S(x_t - \eta \,\text{sign}(\nabla f(x_t))), \tag{2}$$

[1] It is straightforward to extend our method to other constraints bounding the offsets between inputs and adversarial examples.

where sign$(\cdot)$ is a sign function. The main challenge here is how to estimate the gradient $\nabla f(x_t)$ with derivative-free methods, as the network's internal architecture and weights are unknown in the black-box adversarial attack. One technique for doing so is by NES (Wierstra et al., 2008):

$$\nabla f(x_t) \approx \nabla_{x_t} \mathbb{E}_{\mathcal{N}(z|x_t,\sigma^2)} f(z) \quad (3)$$
$$= \mathbb{E}_{\mathcal{N}(z|x_t,\sigma^2)} f(z) \nabla_{x_t} \log \mathcal{N}(z|x_t,\sigma^2), \quad (4)$$

where $\mathcal{N}(z|x_t,\sigma^2)$ is an isometric normal distribution with mean x_t and variance σ^2. Therefore, the stochastic gradient descent (SGD) version of eq. (2) becomes:

$$x_{t+1} \leftarrow \text{proj}_S\big(x_t - \eta \,\text{sign}\big(\frac{1}{b}\sum_{i=1}^{b} f(z_i) \nabla \log \mathcal{N}(z_i|x_t,\sigma^2)\big)\big),$$

where b is the size of a mini-batch and z_i is sampled from the normal distribution. The performance of this approach hinges on the quality of the estimated gradient. Our experiments show that its performance varies on attacking different DNNs probably because non-smooth DNNs lead to unstable NES estimation of the gradients (cf. eq. (3)).

2.2. $\mathcal{N}$ATTACK

We propose a different formulation albeit still motivated by NES. Given an input x and a small region S that contains x (i.e., $S = S_p(x)$ defined earlier), the key idea is to consider a smoothed objective as our optimization criterion:

$$\min_{\theta} \; J(\theta) := \int f(x') \pi_S(x'|\theta) dx' \quad (5)$$

where $\pi_S(x'|\theta)$ is a probability density with support defined on S. Compared with problem (1), this frame assumes that we can find a distribution over S such that the loss $f(x')$ is small in expectation. Hence, a sample drawn from this distribution is likely adversarial. Furthermore, with appropriate $\pi_S(\cdot|\theta)$, the objective $J(\theta)$ is a smooth function of θ, and the optimization process of this formulation does not depend on any estimation of the gradient $\nabla f(x_t)$. Therefore, it is not impeded by the non-smoothness of neural networks. Finally, the distribution over S can be parameterized in a much lower dimensional space ($\dim(\theta) \ll \dim(x)$), giving rise to more efficient algorithms than eq. (2) which directly works in the high-dimensional input space.

2.2.1. The distribution on S

In order to define a distribution $\pi_S(x'|\theta)$ on S, we take the following transformation of variable approach:

$$x' = \text{proj}_S(g(z)), \quad z \sim \mathcal{N}(z|\mu,\sigma^2) \quad (6)$$

where $\mathcal{N}(z|\mu,\sigma^2)$ is an isometric normal distribution whose mean μ and variance σ^2 are *to be learned* and the function $g : \mathbb{R}^{\dim(\mu)} \mapsto \mathbb{R}^{\dim(x)}$ maps a normal instance to the space of the neural network input. We leave it to future work to explore the other types of distributions.

In this work, we implement the transformation of the normal variable by the following steps:

1. draw $z \sim \mathcal{N}(\mu,\sigma^2)$, compute $g(z)$ as
$$g(z) = 1/2(\tanh(g_0(z)) + 1),$$
2. clip $\delta' = \text{clip}_p(g(z) - x)$, $p = 2$ or ∞, and
3. return $\text{proj}_S(g(z))$ as $x' = x + \delta'$

Step 1 draws a "seed" z and then maps it by $g_0(z)$ to the space of the same dimension as the input x. In our experiments, we let z lie in the space of the CIFAR10 images (Krizhevsky & Hinton, 2009) (i.e., $\mathbb{R}^{32\times32\times3}$), so the function $g_0(\cdot)$ is an identity mapping for the experiments on CIFAR10 and a bilinear interpolation for the ImageNet images (Deng et al., 2009). We further transform $g_0(z)$ to the same range as the input by $g(z) = \frac{1}{2}(\tanh(g_0(z)) + 1) \in [0,1]^{\dim(x)}$ and then compute the offset $\delta = g(z) - x$ between the transformed vector and the input. Steps 2 and 3 detail how to project $g(z)$ onto the set S, where the clip functions are respectively

$$\text{clip}_2(\delta) = \begin{cases} \delta\tau_2/\|\delta\|_2 & \text{if } \|\delta\|_2 > \tau_2 \\ \delta & \text{else} \end{cases} \quad (7)$$
$$\text{clip}_\infty(\delta) = \min(\delta, \tau_\infty) \quad (8)$$

with the thresholds τ_2 and τ_∞ given by users.

Thus far, we have fully specified our problem formulation (eq. (5)). Before discussing how to solve this problem, we recall that the set S is the ℓ_p-ball centered at x: $S = S_p(x)$. Since problem (5) is formulated for a particular input to the targeted DNN, the input x also determines the distribution $\pi_S(z|\theta)$ via the dependency of S on x. In other words, we will learn personalized distributions for different inputs.

2.2.2. Optimization

Let $\text{proj}_S(g(z))$ be steps 1–3 in the above variable transformation procedure. We can rewrite the objective function $J(\theta)$ in problem (5) as

$$J(\theta) = \mathbb{E}_{\mathcal{N}(z|\mu,\sigma)} f(\text{proj}(g(z))),$$

where $\theta = (\mu,\sigma^2)$ are the unknowns. We use grid search to find a proper bandwidth σ for the normal distribution and NES to find its mean μ:

$$\mu_{t+1} \leftarrow \mu_t - \eta \nabla_\mu J(\theta)|_{\mu_t}, \quad (9)$$

whose SGD version over a mini-batch of size b is

$$\mu_{t+1} \leftarrow \mu_t - \frac{\eta}{b}\sum_{i=1}^{b} f(\text{proj}_S(g(z_i)))\nabla_\mu \log \mathcal{N}(z_i|\mu_t, \sigma^2).$$

In practice, we sample ϵ_i from a standard normal distribution and then use a linear transformation $z_i = \mu + \epsilon_i\sigma$ to make it follow the distribution $\mathcal{N}(z|\mu, \sigma^2)$. With this notion, we can simplify $\nabla_\mu \log \mathcal{N}(z_i|\mu_t, \sigma^2) \propto \sigma^{-1}\epsilon_i$.

Algorithm 1 summarizes the full algorithm, called $\mathcal{N}$ATTACK, for optimizing our smoothed formulation in eq. (5). In line 6 of Algorithm 1, the z-score operation is to subtract from each loss quantity f_i the mean of the losses $f_1, \cdots, f_b$ and divide it by the standard deviation of all the loss quantities. We find it stablizes $\mathcal{N}$ATTACK; the algorithm converges well with a constant learning rate η. Otherwise, one would have to schedule more sophisticated learning rates as reported in (Ilyas et al., 2018). Regarding the loss function in line 5, we employ the C&W loss (Carlini & Wagner, 2017) in the experiments: $f(x') := \max\left(0, \log F(x')_y - \max_{c\neq y} \log F(x')_c\right)$.

In order to generate an adversarial example for an input x to the neural network classifier $C(\cdot)$, we use the $\mathcal{N}$ATTACK algorithm to find a probability density distribution over $S_p(x)$ and then sample from this distribution until arriving at an adversarial instance x' such that $C(x') \neq C(x)$.

Note that our method differs from that of Ilyas et al. (2018) in that we allow an arbitrary data transformation $g(\cdot)$ which is more flexible than directly seeking the adversarial example in the input space, and we absorb the computation of $\text{proj}_S(\cdot)$ into the function evaluation before the update of μ (line 7 of Algorithm 1). On the contrary, the projection of Ilyas et al. (2018) is after the computation of the estimated gradient (which is similar to line 7 in Algorithm 1) because it is an estimation of the projected gradient. The difference in the computational order of projection is conceptually important because, in our case, the projection is treated as part of the function evaluation, which is more stable than treating it as an estimation of the projected gradient. Practically, this also makes a major difference, which can be seen from our experimental comparisons of the two approaches.

2.3. Initializing $\mathcal{N}$ATTACK by Regression

The initialization to the mean μ_0 in Algorithm 1 plays a key role in terms of run time. When a good initialization is given, we often successfully find adversarial examples in less than 100 iterations. Hence, we propose to boost the $\mathcal{N}$ATTACK algorithm by using a regression neural network. It takes a benign example x as the input and outputs μ_0 to initialize $\mathcal{N}$ATTACK. In order to train this regressor, we generate many (input, adversarial example) pairs $\{(x, x_{adv})\}$ by running $\mathcal{N}$ATTACK on the training set of benchmark datasets. The regression network's weights are then set by minimizing the ℓ_2 loss between the network's output and $g_0^{-1}(\arctan(2x_{adv} - 1)) - g_0^{-1}(\arctan(2x - 1))$; in other words, we regress for the offset between the adversarial example x_{adv} and the input x in the space $\mathbb{R}^{\dim(\mu)}$ of the distribution parameters. The supplementary materials present more details about this regression network.

Algorithm 1 Black-box adversarial $\mathcal{N}$ATTACK

Input: DNN $F(\cdot)$, input x and its label y, initial mean μ_0, standard deviation σ, learning rate η, sample size b, and the maximum number of iterations T

Output: μ_T, mean of the normal distribution

1: **for** $t = 0, 1, ..., T-1$ **do**
2: Sample ϵ_1,...,$\epsilon_b \sim \mathcal{N}(0, I)$
3: Compute $g_i = g(\mu_t + \epsilon_i\sigma)$ by Step 1 $\forall i \in \{1, \cdots, b\}$
4: Obtain proj(g_i) by steps 2–3, $\forall i$
5: Compute losses $f_i := f(\text{proj}(g_i)), \forall i$
6: Z-score $\widehat{f_i} = (f_i - \text{mean}(f))/\text{std}(f), \forall i$
7: Set $\mu_{t+1} \leftarrow \mu_t - \frac{\eta}{b\sigma}\sum_{i=1}^{b} \widehat{f_i}\epsilon_i$
8: **end for**

3. Experiments

We use the proposed $\mathcal{N}$ATTACK to attack 13 defense methods for DNNs published in 2018 or 2019 and two representative vanilla DNNs. For each defense method, we run experiments using the same protocol as reported in the original paper, including the datasets and ℓ_p distance (along with the threshold) to bound the differences between adversarial examples and inputs — this experiment protocol favors the defense method. In particular, CIFAR10 (Krizhevsky & Hinton, 2009) is employed in the attack on nine defense methods and ImageNet (Deng et al., 2009) is used for the remaining four. We examine all the test images of CIFAR10 and randomly choose 1,000 images from the test set of ImageNet. 12 of the defenses concern the ℓ_∞ distance between the adversarial examples and the benign ones and one works with the ℓ_2 distance. We threshold the l_∞ distance in the normalized $[0, 1]^{\dim(x)}$ input space. The l_2 distance is normalized by the number of pixels.

In addition to the main comparison results, we also investigate the defense methods' robustness versus the varying strengths of $\mathcal{N}$ATTACK (cf. Section 3.2). Specifically, we plot the attack success rate versus the attack iteration. The curves provide a complementary metric to the overall attack success rate, uncovering the dynamic traits of the competition between a defense and an attack.

Finally, we examine the adversarial examples' transferabilities between some of the *defended neural networks* (cf. Section 3.3). Results show that, unlike the finding that many adversarial examples are transferable across different

vanilla neural networks, a majority of the adversarial examples that fail one defended DNN cannot defeat the others. In some sense, this weakens the practical significance of white-box attack methods which are often thought applicable to unknown DNN classifiers by attacking a substitute neural network instead (Papernot et al., 2017).

3.1. Attacking 13 Most Recent Defense Techniques

We consider 13 defenses recently developed: adversarial training (ADV-TRAIN) (Madry et al., 2018), adversarial training of Bayesian DNNs (ADV-BNN) (Liu et al., 2019), Thermometer encoding (THERM) (Buckman et al., 2018), THERM-ADV (Athalye et al., 2018; Madry et al., 2018), ADV-GAN (Wang & Yu, 2019), local intrinsic dimensionality (LID) (Ma et al., 2018), stochastic activation pruning (SAP) (Dhillon et al., 2018), random self-ensemble (RSE) (Liu et al., 2018), cascade adversarial training (CAS-ADV) (Na et al., 2018), randomization (Xie et al., 2018), input transformation (INPUT-TRANS) (Guo et al., 2018), pixel deflection (Prakash et al., 2018), and guided denoiser (Liao et al., 2018). We describe them in detail in the supplementary materials. Additionally, we also include Wide Resnet-32 (WRESNET-32) (Zagoruyko & Komodakis, 2016) and INCEPTION V3 (Szegedy et al., 2016), two vanilla neural networks for CIFAR10 and ImageNet, respectively.

Implementation Details. In our experiments, the defended DNNs of SAP, LID, RANDOMIZATION, INPUT-TRANS, THERM, and THERM-DAV come from (Athalye et al., 2018), the defended models of GUIDED DENOISER and PIXEL DEFLECTION are based on (Athalye & Carlini, 2018), and the models defended by RSE, CAS-ADV, ADV-TRAIN, and ADV-GAN are respectively from the original papers. For ADV-BNN, we attack an ensemble of ten BNN models. In all our experiments, we set $T = 600$ as the maximum number of optimization iterations, $b = 300$ for the sample size, variance of the isotropic Gaussian $\sigma^2 = 0.01$, and learning rate $\eta = 0.008$. $\mathcal{N}$ATTACK is able to defeat most of the defenses under this setting and about 90% inputs for other cases. We then fine-tune the learning rate η and sample size b for the hard leftovers.

3.1.1. ATTACK SUCCESS RATES

We report in Table 1 the main comparison results evaluated by the attack success rate, the higher the better. Our $\mathcal{N}$ATTACK achieves 100% success on six out of the 13 defenses and more than 90% on five of the rest. As a single black-box adversarial algorithm, $\mathcal{N}$ATTACK is better than or on par with the set of powerful white-box attack methods of various forms (Athalye et al., 2018), especially on the defended DNNs. It also significantly outperforms three state-of-the-art black-box attack methods: ZOO (Chen et al., 2017), which adopts the zero-th order gradients to find adversarial examples; QL (Ilyas et al., 2018), a query-limited attack based on an evolution strategy; and a decision-based (D-based) attack method (Brendel et al., 2017) mainly generating ℓ_2-bounded adversarial examples.

Notably, ADV-TRAIN is still among the best defense methods, so is its extension to the Bayesian DNNs (i.e., ADV-BNN). However, along with CAS-ADV and THERM-ADV which are also equipped with the adversarial training, their strengths come at the price that they give worse classification performances than the others on the clean inputs (cf. the third column of Table 1). Moreover, ADV-TRAIN incurs extremely high computation cost. When the image resolutions are high, Kurakin et al. (2016) found that it is difficult to run the adversarial training at the ImageNet scale. Since our $\mathcal{N}$ATTACK enables efficient generation of adversarial examples once we learn the distribution, we can potentially scale up the adversarial training with $\mathcal{N}$ATTACK and will explore it in the future work.

We have tuned the main free parameters of the competing methods (e.g., batch size and bandwidth in QL). ZOO runs extremely slow with high-resolution images, so we instead use the hierarchical trick the authors described (Chen et al., 2017) for the experiments on ImageNet. In particular, we run ZOO starting from the attack space of $32 \times 32 \times 3$, lift the resolution to $64 \times 64 \times 3$ after 2,000 iterations and then to $128 \times 128 \times 3$ after 10,000 iterations, and finally up-sample the result to the same size as the DNN input with bilinear interpolation.

3.1.2. ABLATION STUDY AND RUN-TIME COMPARISON

$\mathcal{N}$ATTACK vs. QL. We have discussed the conceptual differences between $\mathcal{N}$ATTACK and QL (Ilyas et al., 2018) in Section 2 (e.g., $\mathcal{N}$ATTACK formulates a smooth optimization criterion and offers a probability density on the ℓ_p-ball of an input). Moreover, the comparison results in Table 1 verify the advantage of $\mathcal{N}$ATTACK over QL in terms of the overall attack strengths. Additionally, we here conduct an ablation study to investigate two major algorithmic differences between them: $\mathcal{N}$ATTACK absorbs the projection (proj_S) into the objective function and allows an arbitrary change of variable transformation $g(\cdot)$. Our study concerns THERM-ADV and SAP, two defended DNNs on which QL respectively reaches 42.3% and 96.2% attack success rates. After we instead absorb the projection in QL into the objective, the results are improved to 54.7% and 97.7%, respectively. If we further apply $g(\cdot)$, the change of variable procedure (cf. Steps 1–3), the success rates become 83.3% and 98.9%, respectively. Finally, with the z-score operation (line 6 of Algorithm 1), the results are boosted to 90.9%/100%, approaching $\mathcal{N}$ATTACK's 91.2%/100%. Therefore, we say that $\mathcal{N}$ATTACK boosts QL's performance, thanks to both the smoothed objective and the transformation $g(\cdot)$.

$\mathcal{N}$ATTACK vs. the White-Box BPDA Attack. While BPDA achieves high attack success rates by different vari-

Table 1. Adversarial attack on 13 recently published defense methods. (* the number reported in (Athalye et al., 2018). For all the other numbers, we obtain them by running the code released by the authors or implemented ourselves with the help of the authors. For D-based and ADV-TRAIN, we respectively report the results on 100 and 1000 images only because they incur expensive computation costs.)

Defense Technique	Dataset	Classification Accuracy %	Threshold & Distance	Attack Success Rate %				
				BPDA	ZOO	QL	D-based	$\mathcal{N}$ATTACK
ADV-TRAIN (Madry et al., 2018)	CIFAR10	87.3	0.031 (L_∞)	46.9	16.9	40.3	–	**47.9**
ADV-BNN (Liu et al., 2019)	CIFAR10	79.7	0.035 (L_∞)	48.3	–	–	–	**75.3**
THERM-ADV (Athalye et al., 2018)	CIFAR10	88.5	0.031 (L_∞)	76.1	0.0	42.3	–	**91.2**
CAS-ADV (Na et al., 2018)	CIFAR10	75.6	0.015 (L_∞)	85.0*	96.1	68.4	–	**97.7**
ADV-GAN (Wang & Yu, 2019)	CIFAR10	90.9	0.031 (L_∞)	48.9	76.4	53.7	–	**98.3**
LID (Ma et al., 2018)	CIFAR10	66.9	0.031 (L_∞)	95.0	92.9	95.7	–	**100.0**
THERM (Buckman et al., 2018)	CIFAR10	92.8	0.031 (L_∞)	**100.0**	0.0	96.5	–	**100.0**
SAP (Dhillon et al., 2018)	CIFAR10	93.3	0.031 (L_∞)	**100.0**	5.9	96.2	–	**100.0**
RSE (Liu et al., 2018)	CIFAR10	91.4	0.031 (L_∞)	–	–	–	–	**100.0**
VANILLA WRESNET-32 (Zagoruyko & Komodakis, 2016)	CIFAR10	95.0	0.031 (L_∞)	**100.0**	99.3	96.8	–	**100.0**
GUIDED DENOISER (Liao et al., 2018)	ImageNet	79.1	0.031 (L_∞)	**100.0**	–	–	–	95.5
RANDOMIZATION (Xie et al., 2018)	ImageNet	77.8	0.031 (L_∞)	**100.0**	6.7	45.9	–	96.5
INPUT-TRANS (Guo et al., 2018)	ImageNet	77.6	0.05 (L_2)	**100.0**	38.3	66.5	66.0	**100.0**
PIXEL DEFLECTION (Prakash et al., 2018)	ImageNet	69.1	0.015 (L_∞)	97.0	–	8.5	–	**100.0**
VANILLA INCEPTION V3 (Szegedy et al., 2016)	ImageNet	78.0	0.031 (L_∞)	**100.0**	62.1	100.0	–	**100.0**

ants for handling the diverse defense techniques, $\mathcal{N}$ATTACK gives rise to better or comparable results by a single universal algorithm. Additionally, we compare them in terms of the run time in the supplementary materials; the main observations are the following. On CIFAR10, BPDA and $\mathcal{N}$ATTACK can both find an adversarial example in about 30s. To defeat an ImageNet image, it takes $\mathcal{N}$ATTACK about 71s without the regression network and 48s when it is equipped with the regression net; in contrast, BPDA only needs 4s. It is surprising to see that BPDA is almost 7 times faster at attacking a DNN for ImageNet than a DNN for CIFAR10. It is probably because the gradients of the former are not "obfuscated" as well as the latter due to the higher resolution of the ImageNet input.

3.2. Attack Success Rate vs. Attack Iteration

The $\mathcal{N}$ATTACK algorithm has an appealing property as follows. In expectation, the loss (eq. (5)) decreases at every iteration and hence a sample drawn from the distribution $\pi_S(x|\theta)$ is adversarial with higher chance. Though there could be oscillations, we find that the attack strengths do grow monotonically with respect to the evolution iterations in our experiments. Hence, we propose a new curve shown in Figure 1a featuring the attack success rate versus number of evolution iterations — strength of attack. For the experiment here, the Gaussian mean μ_0 is initialized by $\mu_0 \sim \mathcal{N}(g_0^{-1}(\arctan(2x-1)), \sigma^2)$ for any input x to maintain about the same starting points for all the curves.

Figure 1a plots eight defense methods on CIFAR10 along with a vanilla DNN. It is clear that ADV-TRAIN, ADV-BNN, THERM-ADV, and CAS-ADV, which all employ the adversarial training strategy, are more difficult to attack than the others. What's more interesting is with the other five DNNs. Although $\mathcal{N}$ATTACK completely defeats them all by the end, the curve of the vanilla DNN is the steepest while the SAP curve rises much slower. If there are constraints on the computation time or the number of queries to the DNN classifiers, SAP is advantageous over the vanilla DNN, RSE, THERM, and LID.

Note that the ranking of the defenses in Table 1 (evaluation

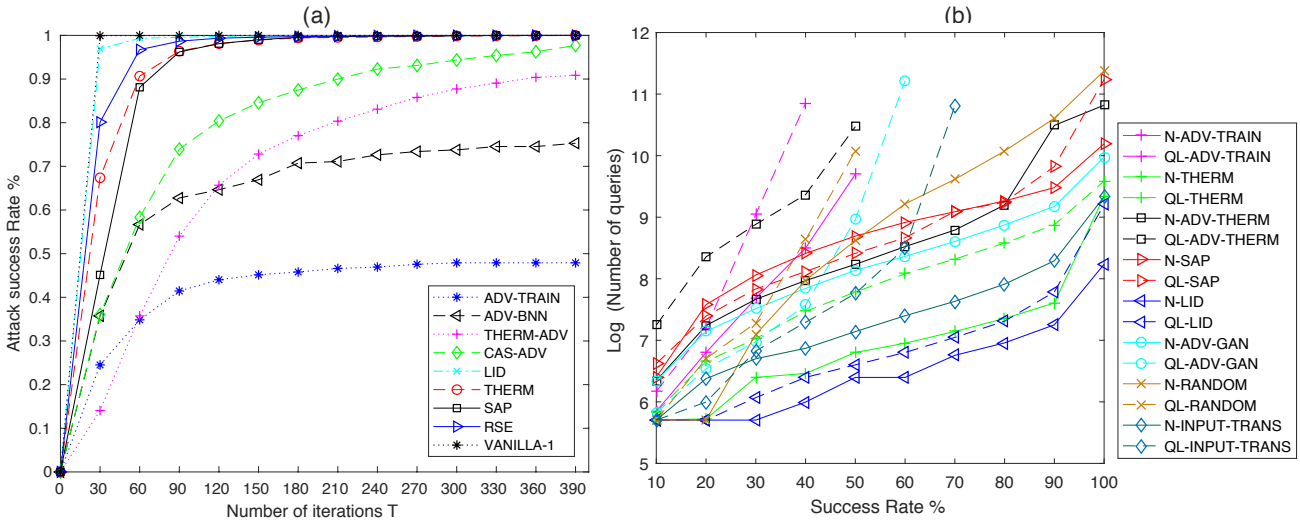

Figure 1. (a) Success rate versus run steps of $\mathcal{N}$ATTACK. (b) Comparison results with QL measured by the log of average number of queries per successful image. The solid lines denote $\mathcal{N}$ATTACK and the dashed lines illustrate QL.

by the success rate) is different from the ordering on the left half of Figure 1a, signifying the attack success rate and the curve mutually complement. The curve reveals more characteristics of the defense methods especially when there are constraints on the computation time or number of queries to the DNN classifier.

Figure 1b shows $\mathcal{N}$ATTACK (solid lines) is more query efficient than the QL attack (Ilyas et al., 2018) (dashed lines) on 6 defenses under most attack success rates and the difference is even amplified for higher success rates. For SAP, $\mathcal{N}$ATTACK performs better when the desired attack success rate is bigger than 80%.

3.3. Transferability

We also study the transferability of adversarial examples across different *defended* DNNs. This study differs from the earlier ones on *vanilla* DNNs (Szegedy et al., 2013; Liu et al., 2016). We investigate both the white-box attack BPDA and our black-box $\mathcal{N}$ATTACK.

Following the experiment setup in (Kurakin et al., 2016), we randomly select 1000 images for each targeted DNN such that they are classified correctly, and yet the adversarial images of them are classified incorrectly. We then use the adversarial examples of the 1000 images to attack the other DNNs. In addition to the defended DNNs, we also include two vanilla DNNs for reference: VANILLA-1 and VANILLA-2. VANILLA-1 is a light-weight DNN classifier built by (Carlini & Wagner, 2017) with 80% accuracy on CIFAR10. VANILLA-2 is the Wide-ResNet-28 (Zagoruyko & Komodakis, 2016) which gives rise to 92.3% classification accuracy on CIFAR10. For fair comparison, we change the threshold τ_∞ to 0.031 for CAS-ADV. We exclude RSE and CAS-ADV from BPDA's confusion table because it is not obviously clear how to attack RSE using BPDA and the released BPDA code lacks the piece for attacking CAS-ADV.

The confusion tables of BPDA and $\mathcal{N}$ATTACK are shown in Figure 2, respectively, where each entry indicates the success rate of using the adversarial examples originally targeting the row-wise defense model to attack the column-wise defense. Both confusion tables are asymmetric; it is easier to transfer from defended models to the vanilla DNNs than vice versa. Besides, the overall transferrability is lower than that across the DNNs without any defenses (Liu et al., 2016). We highlight some additional observations below.

Firstly, the transferability of our black-box $\mathcal{N}$ATTACK is not as good as the black-box BPDA attack. This is probably because BPDA is able to explore the intrinsically common part of the DNN classifiers — it has the privilege of accessing the true or estimated gradients that observe the DNNs' architectures and weights.

Secondly, both the network architecture and defense methods can influence the transferability. VANILLA-2 is the underlying classifier of SAP, THERM-ADV, and THERM. The adversarial examples originally attacking VANILLA-2 do transfer better to SAP and THERM than to the others probably because they share the same DNN architecture, but the examples achieve very low success rate on THERM-ADV due to the defense technique.

Finally, the transfer success rates are low no matter from THERM-ADV to the other defenses or vice versa, and ADV-TRAIN and ADV-BNN lead to fairly good results of transfer attacks on the other defenses and yet themselves are robust against the adversarial examples of the other defended DNNs. The unique result of THERM-ADV probably attributes to its use of double defense techniques, i.e., Thermometer encoding and adversarial training.

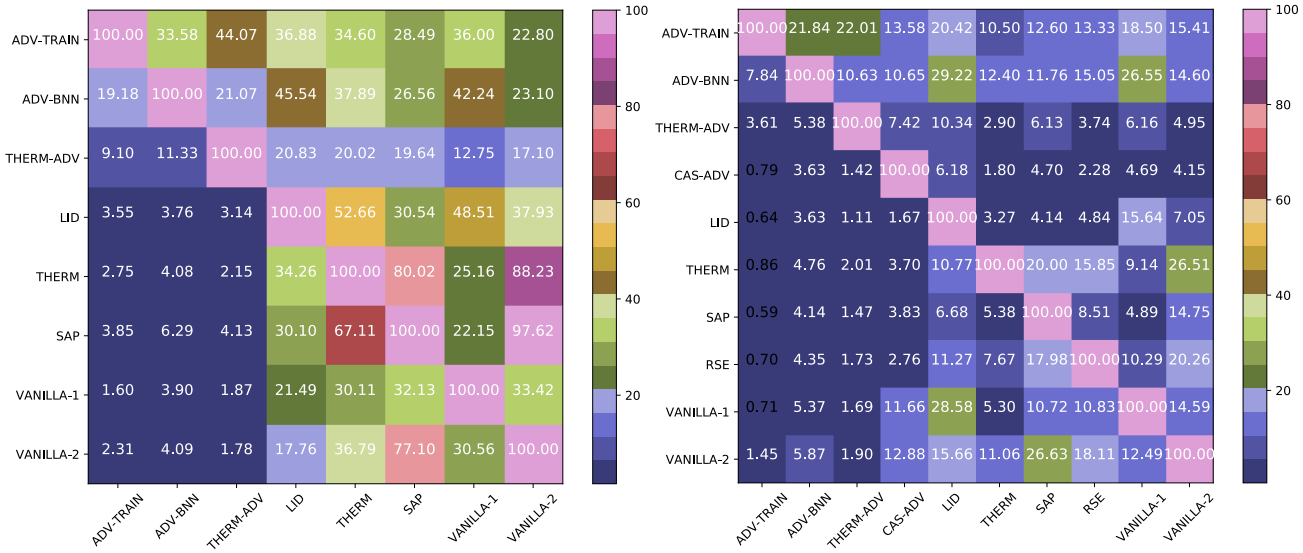

Figure 2. Transferabilities of BPDA (Athalye et al., 2018) (left) and $\mathcal{N}$ATTACK (right). Each entry shows the attack success rate of attacking the column-wise defense by the adversarial examples that are originally generated for the row-wise DNN.

4. Related Work

There is a vast literature of adversarial attacks on and defenses for DNNs. We focus on the most related works in this section rather than a thorough survey.

White-Box Attacks. The adversary has full access to the target DNN in the white-box attack. Szegedy et al. (2013) first find that DNNs are fragile to the adversarial examples by using box-constrained L-BFGS. Goodfellow et al. (2014a) propose a fast gradient sign (*FGS*) method, which is featured by efficiency and high performance for generating the ℓ_∞ bounded adversarial examples. Papernot et al. (2016b) and Moosavi-Dezfooli et al. (2016) instead formulate the problems with the l_0 and ℓ_2 metrics, respectively. (Carlini & Wagner, 2017) have proposed a powerful iterative optimization based attack. Similarly, a projected gradient descent has been shown strong in attacking DNNs (Madry et al., 2018). Most the white-box attacks rely on the gradients of the DNNs. When the gradients are "obfuscated" (e.g., by randomization), (Athalye et al., 2018) derive various methods to approximate the gradients, while we use a single algorithm to attack a variety of defended DNNs.

Black-Box Attacks. As the name suggests, some parts of the DNNs are treated as black boxes in the black-box attack. Thanks to the adversarial examples' transferabilities (Szegedy et al., 2013), Papernot et al. (2017) train a substitute DNN to imitate the target black-box DNN, produce adversarial examples of the substitute model, and then use them to attack the target DNN. Chen et al. (2017) instead use the zero-th order optimization to find adversarial examples. Ilyas et al. (2018) use the evolution strategy (Salimans et al., 2017) to approximate the gradients. Brendel et al. (2017) introduce a decision-based attack by reading the hard labels predicted by a DNN, rather than the soft probabilistic output. Similarly, Cheng et al. (2019) also provide a formulation to explore the hard labels. Most of the existing black-box methods are tested against vanilla DNNs. In this work, we test them on defended ones along with our $\mathcal{N}$ATTACK.

5. Conclusion and Future Work

In this paper, we present a black-box adversarial attack method which learns a probability density on the ℓ_p-ball of a clean input to the targeted neural network. One of the major advantages of our approach is that it allows an arbitrary transformation of variable $g(\cdot)$, converting the adversarial attack to a space of much lower dimensional than the input space. Experiments show that our algorithm defeats 13 defended DNNs, better than or on par with state-of-the-art white-box attack methods. Additionally, our experiments on the transferability of the adversarial examples across the defended DNNs show different results reported in the literature: unlike the high transferability across vanilla DNNs, it is difficult to transfer the attacks on the defended DNNs.

Some existing works try to characterize the adversarial examples by their geometric properties. In contrast to this macro view, we model the adversarial population of each single input from a micro view by a probabilistic density. There are still a lot to explore along this avenue. *What is a good family of distributions to model the adversarial examples? How to conduct adversarial training by efficiently sampling from the distribution?* These questions are worth further investigation in the future work.

Acknowledgement: This work was supported in part by NSF-1836881, NSF-1741431, and ONR-N00014-18-1-2121.

References

Athalye, A. and Carlini, N. On the robustness of the cvpr 2018 white-box adversarial example defenses. *arXiv preprint arXiv:1804.03286*, 2018.

Athalye, A., Carlini, N., and Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

Brendel, W., Rauber, J., and Bethge, M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

Buckman, J., Roy, A., Raffel, C., and Goodfellow, I. Thermometer encoding: One hot way to resist adversarial examples. 2018.

Carlini, N. and Wagner, D. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pp. 39–57. IEEE, 2017.

Chen, P.-Y., Zhang, H., Sharma, Y., Yi, J., and Hsieh, C.-J. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 15–26. ACM, 2017.

Cheng, C.-H., Nührenberg, G., and Ruess, H. Maximum resilience of artificial neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pp. 251–268. Springer, 2017.

Cheng, M., Le, T., Chen, P.-Y., Zhang, H., Yi, J., and Hsieh, C.-J. Query-efficient hard-label black-box attack: An optimization-based approach. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJlk6iRqKX.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255. IEEE, 2009.

Dhillon, G. S., Azizzadenesheli, K., Lipton, Z. C., Bernstein, J., Kossaifi, J., Khanna, A., and Anandkumar, A. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88 (2):303–338, June 2010.

Fawzi, A., Moosavi-Dezfooli, S.-M., Frossard, P., and Soatto, S. Empirical study of the topology and geometry of deep networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Fischetti, M. and Jo, J. Deep neural networks as 0-1 mixed integer linear programs: A feasibility study. *arXiv preprint arXiv:1712.06174*, 2017.

Gan, C., Li, Y., Li, H., Sun, C., and Gong, B. Vqs: Linking segmentations to questions and answers for supervised attention in vqa and question-focused semantic segmentation. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014a.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014b.

Guo, C., Rana, M., Cisse, M., and van der Maaten, L. Countering adversarial images using input transformations. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SyJ7ClWCb.

He, W., Wei, J., Chen, X., Carlini, N., and Song, D. Adversarial example defenses: Ensembles of weak defenses are not strong. *arXiv preprint arXiv:1706.04701*, 2017.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., and Kingsbury, B. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012.

Ilyas, A., Engstrom, L., Athalye, A., and Lin, J. Black-box adversarial attacks with limited queries and information. *arXiv preprint arXiv:1804.08598*, 2018.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images, 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

Li, Y., Wang, L., Yang, T., and Gong, B. How local is the local diversity? reinforcing sequential determinantal point processes with dynamic ground sets for supervised video summarization. In *The European Conference on Computer Vision (ECCV)*, September 2018.

Liao, F., Liang, M., Dong, Y., Pang, T., Zhu, J., and Hu, X. Defense against adversarial attacks using high-level representation guided denoiser. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1778–1787, 2018.

Liu, X., Cheng, M., Zhang, H., and Hsieh, C.-J. Towards robust neural networks via random self-ensemble. In *European Conference on Computer Vision*, pp. 1–8. Springer, 2018.

Liu, X., Li, Y., Wu, C., and Hsieh, C.-J. Adv-BNN: Improved adversarial defense through robust bayesian neural network. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rk4Qso0cKm.

Liu, Y., Chen, X., Liu, C., and Song, D. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

Lomuscio, A. and Maganti, L. An approach to reachability analysis for feed-forward relu neural networks. *arXiv preprint arXiv:1706.07351*, 2017.

Ma, X., Li, B., Wang, Y., Erfani, S. M., Wijewickrema, S., Houle, M. E., Schoenebeck, G., Song, D., and Bailey, J. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*, 2018.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2574–2582, 2016.

Moosavi-Dezfooli, S.-M., Fawzi, A., Fawzi, O., and Frossard, P. Universal adversarial perturbations. *arXiv preprint*, 2017.

Na, T., Ko, J. H., and Mukhopadhyay, S. Cascade adversarial machine learning regularized with a unified embedding. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HyRVBzap-.

Papernot, N., McDaniel, P., Wu, X., Jha, S., and Swami, A. Distillation as a defense to adversarial perturbations against deep neural networks. *arXiv preprint arXiv:1511.04508*, 2015.

Papernot, N., McDaniel, P., and Goodfellow, I. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016a.

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., and Swami, A. The limitations of deep learning in adversarial settings. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pp. 372–387. IEEE, 2016b.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pp. 506–519. ACM, 2017.

Prakash, A., Moran, N., Garber, S., DiLillo, A., and Storer, J. Deflecting adversarial attacks with pixel deflection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8571–8580, 2018.

Rozsa, A., Gunther, M., and Boult, T. E. Towards robust deep neural networks with bang. *arXiv preprint arXiv:1612.00138*, 2016.

Salimans, T., Ho, J., Chen, X., and Sutskever, I. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

Sharma, Y. and Chen, P.-Y. Breaking the madry defense model with l_1-based adversarial examples. *arXiv preprint arXiv:1710.10733*, 2017.

Shelhamer, E., Long, J., and Darrell, T. Fully convolutional networks for semantic segmentation. *arXiv preprint arXiv:1605.06211*, 2016.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017a.

Tramèr, F., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. The space of transferable adversarial examples. *arXiv preprint arXiv:1704.03453*, 2017b.

Tsuzuku, Y., Sato, I., and Sugiyama, M. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. *arXiv preprint arXiv:1802.04034*, 2018.

Wang, H. and Yu, C.-N. A direct approach to robust deep learning using adversarial networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1lIMn05F7.

Weng, T.-W., Zhang, H., Chen, H., Song, Z., Hsieh, C.-J., Boning, D., Dhillon, I. S., and Daniel, L. Towards fast computation of certified robustness for relu networks. *arXiv preprint arXiv:1804.09699*, 2018.

Wierstra, D., Schaul, T., Peters, J., and Schmidhuber, J. Natural evolution strategies. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*, pp. 3381–3387. IEEE, 2008.

Wong, E. and Kolter, Z. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pp. 5283–5292, 2018.

Xie, C., Wang, J., Zhang, Z., Ren, Z., and Yuille, A. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Sk9yuql0Z.

Xu, K., Liu, S., Zhang, G., Sun, M., Zhao, P., Fan, Q., Gan, C., and Lin, X. Interpreting adversarial examples by activation promotion and suppression. *CoRR*, abs/1904.02057, 2019. URL http://arxiv.org/abs/1904.02057.

Xu, W., Evans, D., and Qi, Y. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

Zagoruyko, S. and Komodakis, N. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Zhang, H., Weng, T.-W., Chen, P.-Y., Hsieh, C.-J., and Daniel, L. Efficient neural network robustness certification with general activation functions. In *Advances in Neural Information Processing Systems (NuerIPS)*, dec 2018.

Supplementary Materials for $\mathcal{N}$ATTACK: Learning the Distributions of Adversarial Examples for an Improved Black-Box Attack on Deep Neural Networks

In this supplementary document, we provide the following details to support the main text:

Section A: descriptions of the 13 defense methods studied in the experiments,

Section B: architecture of the regression neural network for initializing our $\mathcal{N}$ATTACK algorithm, and

Section C: run-time analysis about $\mathcal{N}$ATTACK and BPDA (Athalye et al., 2018).

A. More Details of the 13 Defense Methods

- **Thermometer encoding (THERM).** To break the hypothesized linearity behavior of DNNs (Goodfellow et al., 2014a), Buckman et al. (2018) proposed to transform the input by non-differentiable and non-linear thermometer encoding, followed by a slight change to the input layer of conventional DNNs.

- **ADV-TRAIN & THERM-ADV.** Madry et al. (2018) proposed a defense using adversarial training (ADV-TRAIN). Specially, the training procedure alternates between seeking an "optimal" adversarial example for each input by projected gradient descent (PGD) and minimizing the classification loss under the PGD attack. Furthermore, Athalye et al. (2018) find that the adversarial robust training (Madry et al., 2018) can significantly improve the defense strength of THERM (THERM-ADV). Compared with ADV-TRAIN, the adversarial examples are produced by the logit-space projected gradient ascent in the training.

- **Cascade adversarial training (CAS-ADV).** Na et al. (2018) reduced the computation cost of the adversarial training (Goodfellow et al., 2014b; Kurakin et al., 2016) in a cascade manner. A model is trained from the clean data and one-step adversarial examples first. The second model is trained from the original data, one-step adversarial examples, as well as iterative adversarial examples generated against the first model. Additionally, a regularization is introduced to the unified embeddings of the clean and adversarial examples.

- **Adversarially trained Bayesian neural network (ADV-BNN).** Liu et al. (2019) proposed to model the randomness added to DNNs in a Bayesian framework in order to defend against adversarial attack. Besides, they incorporated the adversarial training, which has been shown effective in the previous works, into the framework.

- **Adversarial training with adversarial examples generated from GAN (ADV-GAN).** Wang & Yu (2019) proposed to model the adversarial perturbation with a generative network, and they learned it jointly with the defensive DNN as a discriminator.

- **Stochastic activation pruning (SAP).** Dhillon et al. (2018) randomly dropped some neurons of each layer with the probabilities in proportion to their absolute values.

- **RANDOMIZATION.** (Xie et al., 2018) added a randomization layer between inputs and a DNN classifier. This layer consists of resizing an image to a random resolution, zero-padding, and randomly selecting one from many resulting images as the actual input to the classifier.

- **Input transformation (INPUT-TRANS).** By a similar idea as above, Guo et al. (2018) explored several combinations of input transformations coupled with adversarial training, such as image cropping and rescaling, bit-depth reduction, JPEG compression.

- **PIXEL DEFLECTION.** Prakash et al. (2018) randomly sample a pixel from an image and then replace it with another pixel randomly sampled from the former's neighborhood. Discrete wavelet transform is also employed to filter out adversarial perturbations to the input.

- **GUIDED DENOISER.** Liao et al. (2018) use a denoising network architecture to estimate the additive adversarial perturbation to an input.

- **Random self-ensemble (RSE).** Liu et al. (2018) combine the ideas of randomness and ensemble using the same underlying neural network. Given an input, it generates an ensemble of predictions by adding distinct noises to the network multiple times.

Table 2. Average run time to find an adversarial example ($\mathcal{N}$ATTACK-R stands for $\mathcal{N}$ATTACK initialized with the regression net).

Defense	Dataset	BPDA (Athalye et al., 2018)	$\mathcal{N}$ATTACK	$\mathcal{N}$ATTACK-R
SAP (Dhillon et al., 2018)	CIFAR-10 (L_∞)	33.3s	29.4s	–
RANDOMIZATION (Xie et al., 2018)	ImageNet (L_∞)	3.51s	70.77s	48.22s

B. Architecture of the Regression Network

We construct our regression neural network by using the fully convolutional network (FCN) architecture (Shelhamer et al., 2016). In particular, we adapt the FCN model pre-trained on PASCAL VOC segmentation challenge (Everingham et al., 2010) to our work by changing its last two layers, such that the network outputs an adversarial perturbation of the size $32 \times 32 \times 3$. We train this network by a mean square loss.

C. Run Time Comparison

Compared with the white-box attack approach BPDA (Athalye et al., 2018), $\mathcal{N}$ATTACK may take longer time since BPDA can find the local optimal solution quickly being guided by the approximate gradients. However, $\mathcal{N}$ATTACK can be executed in parallel in each episode. We leave implement the parallel version of our algorithm to the future work and compare its sing-thread version with BPDA below.

We attack 100 samples on one machine with fou TITAN-XP graphic cards and calculate the average run time for reaching an adversarial example. As shown in Table 2, $\mathcal{N}$ATTACK can succeed even faster than the white-box BPDA on CIFAR-10, yet runs slower on ImageNet. The main reason is that when the image size is as small as CIFAR10 (3*32*32), the search space is moderate. However, the run time could be lengthy for high resolution images like ImageNet (3*299*299) especially for some hard cases (we can find the adversarial examples for nearly 90% test images but it could take about 60 minutes for a hard case).

We use a regression net to approximate a good initialization of μ_0 and we name $\mathcal{N}$ATTACK initialized with the regression net as $\mathcal{N}$ATTACK-R. We run $\mathcal{N}$ATTACK and $\mathcal{N}$ATTACK-R on ImageNet with the mini-batch size $b = 40$. The success rate for $\mathcal{N}$ATTACK with random initialization is 82% and for $\mathcal{N}$ATTACK-R is 91.9%, verifying the efficacy of the regression net. The run time shown in Table 2 is calculated on the images with successful attacks. The results demonstrate that $\mathcal{N}$ATTACK-R can reduce by 22.5s attack time per image compared with the random initialization.